Exercícios sobre Programação em EViews

Resolução: Para resolver na aula de 12/03

1. Começe por criar um novo workfile com $n = 6$ observações sem datas específicas. Agora construa as seguintes matrizes/vectores:

$$X = \begin{bmatrix} 1.5 & 0.5 \\ 0.7 & 2.2 \\ 1 & 1.3 \\ 2 & 1.7 \\ 2.5 & 4 \\ 3 & 3.5 \end{bmatrix} \qquad Y = \begin{bmatrix} 15 \\ 17 \\ 18.2 \\ 12.3 \\ 13.4 \\ 14 \end{bmatrix} \qquad \beta = \begin{bmatrix} 1 \\ 0.5 \\ 2 \end{bmatrix}$$

$$U = (u_1, \ldots, u_6) \text{ em que } u_i \overset{i.i.d.}{\sim} N(0,1)$$

$$\nu = (\nu_1, \ldots, \nu_6) \text{ em que } \nu_i \overset{i.i.d.}{\sim} N(4, 2^2)$$

```
!n = 6
workfile test u 1 !n
matrix(!n,2) x
x.fill 1.5, 0.7, 1, 2, 2.5, 3, 0.5, 2.2, 1.3, 1.7, 4, 3.5
vector(!n) y
y.fill 15, 17,18.2, 12.3, 13.4, 14
vector(3) beta
beta.fill 1, 0.5, 2
matrix u=@mnrnd(!n,1)
!mu_v = 4
!sigma_v = 2
matrix v=mu_v+sigma_v*@mnrnd(!n,1)
```

2. Construa os vetores x_1, x_2 e x_3 com um vetor de 1s, a primeira e a segunda coluna da matriz X, respetivamente. Modifique a matriz X de modo a conter o vetor de 1s na primeira coluna. Ainda use a matriz X modificada para construir a seguinte matriz:

$$Xsub = \begin{bmatrix} 0.7 & 2.2 \\ 1 & 1.3 \\ 2 & 1.7 \end{bmatrix}$$

Finalmente use os vetores x_1, x_2 e x_3 para construir novamente a matriz X.

```
vector const = 2*@ones(!n,1) ou const = 1
vector x1 = const
vector x2 = @columnextract(x,1)
vector x3 = @columnextract(x,2)
x = @hcat(const,x)
matrix xsub = @subextract(x,2,2,4,3)
matrix x = @hcat(x1,x2)
matrix x = @hcat(x,x3)
x= 0
matplace(x,x1,1,1)
matplace(x,x2,1,2)
matplace(x,x3,1,3)
```

3. Converta a matriz X e os vetores x1, x2, x3 e y em séries.

```
mtos(y,ys)
mtos(x,xs)
mtos(u,us)
mtos(x1,x1s)
mtos(x2,x2s)
mtos(x3,x3s)
```

4. Obtenha algumas estatísticas descritivas dos vetores/séries da matrix X.

```
freeze(xstat) x.stats
group g1 x1s x2s x3s
freeze(gseries) g1.line(m)
vector qx = @quantile(x3s,0.9)
vector mx =@cmean(x)
vector maxx = @cmax(x)
vector minx = @cmin(x)
```

5. Estime o modelo de regressão linear assumindo que Y e X são a variável dependente e os regressores, respetivamente. Calcule os estimadores OLS, valores ajustados, resíduos, s2, erros padrão dos estimadores, estatísticas t e valores-p usando dois métodos:

   (a) Cálculo manual

```
vector beta = @inverse(@transpose(x)*x)*@transpose(x)*y
vector res = y-x*beta
scalar ssqr = @transpose(y-x*beta)*(y-x*beta)
ou
vector res2 = @emult(y-x*beta, y-x*beta)
ou
vector res2 = @epow(y-x*beta,2)
scalar s2 = ssqr/(@rows(x)-@columns(x))
ou
scalar s2_1 = @sum(res2)/(@rows(x)-@columns(x))
matrix stderror_beta = @sqrt(@getmaindiagonal(s2*@inverse(@transpose(x)*x)))
vector tstatistic = @ediv(beta,stderror_beta)
vector pvalues = 2*(@ones(@columns(x))-@ctdist(@abs(tstatistic),@rows(x)-
@columns(x)))
```

(b) Usando o ambiente "equation"

```
equation eq1.ls ys x1s x2s x3s ou equation eq1.ls ys xs
vector beta = @coefs
series res = resid
eq1.fit fitted_v
scalar ssqr = @ssr
scalar s2 = @ssr/(@rows(x)-@columns(x))
matrix stderror_beta = @stderrs
vector tstatistic = @tstats
vector pvalues = @pvals
```

Compare os resultados obtidos

6. Crie uma matriz diagonal usando *loops*. Crie também outra matriz diagonal mas com falhas do tipo:

$$\begin{bmatrix} 2 & 0 & \cdots & 0 \\ 0 & 0 & & \vdots \\ \vdots & & \ddots & 0 \\ 0 & 0 & \cdots & 2 \end{bmatrix}$$

```
!imax = 5
matrix(!imax,!imax) amat = 0
for !i=1 to !imax
amat(!i,!i) = !i
next
!imax = 5
matrix(!imax,!imax) amat = 0
for !i=1 to !imax step = 2
amat(!i,!i) = !i
next
```

7. Crie uma matriz do tipo usando *loops*:

$$\begin{bmatrix} 1 & 0 & \cdots & 0 \\ 2 & 1 & & \vdots \\ \vdots & & \ddots & 0 \\ 6 & 5 & \cdots & 1 \end{bmatrix}$$

usando *loops*

```
!imax = 5
matrix(!imax,!imax) bmat = 0
for !i=1 to !imax
for !j=!i to !imax
bmat(!j,!i) = !j-!i+1
next
next
```

8. Obtenha uma amostra simulada de 100 observações de um processo AR(1), ARMA(1,1) e AR(2) usando os comandos "for" e "if". Escolha os parâmetros a seu gosto.

```
!n = 100
!rho = 0.8
vector(!n) u = @mnrnd(!n,1)
vector(!n) y = NA
for !i=1 to !n
if !i = 1 then
y(!i) = u(!i)
else
y(!i) = !rho*y(!i-1)+v(!i)
endif
next
mtos(y,ys)
```

9. Importe os dados no ficheiro "data.xls" para uma matriz. Agora obtenha os coeficientes do modelo de regressão linear usando loops:

$$\Delta log(GDP_t) = \beta_0 + \beta_1 TBILL_t + u_t$$

de forma recursiva, ou seja, usando as observações $t = 1, \ldots, T^*$ para $T^* = 2, 3, 4, \ldots, T$.

```
workfile eviews_programming2 q 1947 2004
matrix(232,2) mat1
mat1.read(b2) "C: \ Users \ nsobreira \ Desktop\ temp\ data.xls"
vector gdp=@columnextract(mat1,1)
vector tbill =@columnextract(mat1,2)
mtos(gdp,gdps)
mtos(tbill,tbills)
delete gdp
delete tbill
series gdp = gdps
series tbill = tbills
delete gdps
delete tbills
!bigt = @rows(mat1)
matrix(!bigt,2) coefs_mat1 = NA
!imin = 2
for !i=!imin to !bigt
smpl @first @first+!i
equation eq1.ls dlog(gdp) c tbill
coefs_mat1(!i,1) =@coefs(1)
coefs_mat1(!i,2) =@coefs(2)
next
smpl @all
```

10. Obtenha os coeficientes do modelo de regressão linear usando loops:

$$\Delta log(GDP_t) = \beta_0 + \beta_1 TBILL_t + u_t$$

de forma recursiva mas agora usando um número de observações fixo ("janela fixa") em cada iteração. Use uma janela de 50 observações.

```
!imin = 1
!window = 50
matrix(!bigt-!window,2) coefs_mat2 = NA
for !i=!imin to !bigt-!window
smpl @first+!i-1 @first+!i-1+!window
equation eq1.ls dlog(gdp) c tbill
coefs_mat2(!i,1) =@coefs(1)
coefs_mat2(!i,2) =@coefs(2)
next
smpl @all
```

11. Avalie o enviesamento do estimador OLS de ρ em amostras finitas para um processo AR(1) através de simulações Monte Carlo. Estude também as distorções em amostras finitas do nível de significância da estatística teste-t para $H_0 : \rho = \rho_0$ vs $\rho \neq \rho_0$. Utilize 100, 200 observações e 1000, 10000 replicações. Tente diferentes valores de ρ. Experimente também usar os erros padrão HAC. Utilize rndseed 1234567.

```
rndseed 1234567
!n = 200
workfile eviews_programming3 u 1 !n
vector(5,1) rho_grid
rho_grid.fill 0.2, 0.4, 0.6, 0.8, 1
!phi = 0
!nrep = 10000
matrix(!nrep,@rows(rho_grid)) est_rho_bias = NA
matrix(!nrep,@rows(rho_grid)) tstat_mat = NA
for !rho_case=1 to @rows(rho_grid)
!rho = rho_grid(!rho_case)
!rho_0 = !rho
for !rep = 1 to !nrep
vector(!n) u = @mnrnd(!n,1)
vector(!n) v = NA
vector(!n) y = NA
for !i=1 to !n
if !i = 1 then
v(!i) = u(!i)
y(!i) = v(!i)
else
v(!i) = !phi*v(!i-1)+u(!i)
y(!i) = !rho*y(!i-1)+v(!i)
endif
next
mtos(y,ys)
equation eq1.ls(cov=hac) ys c ar(1)
est_rho_bias(!rep,!rho_case) = @abs(@coefs(2)-!rho_0)
!tstat = (@coefs(2)-!rho)/@stderrs(2)
tstat_mat(!rep,!rho_case) = abs(!tstat)>@qnorm(0.975)
next
next
vector rejH0 = @cmean(tstat_mat)
```